Deep generative models of natural images

Emily Denton

Spring 2016

Outline

Generative models

- Have access to $x \sim p_{data}(x)$ through training set
- Want to learn a model $x \sim p_{model}(x)$
- Want p_{model} to be similar to p_{data}
  - Samples drawn from p_{model} reflect structure of p_{data}
  - Samples from true data distribution have high likelihood under p_{model}

Why do generative modeling?

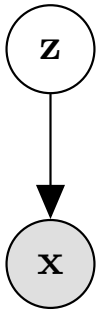
- Unsupervised representation learning
  - Can transfer learned representation so discriminative tasks, retrieval, clustering, etc.
- Train network with both discriminative and generative criterion
  - Utilize unlabeled data, regularize
- Understand data
- Density estimation
- Data augmentation
- ...

Focus of this talk

Generative modeling is a HUGE field...I will focus on (a selected set of) deep directed models of natural images

Outline

Directed graphical models



- We assume data is generated by:

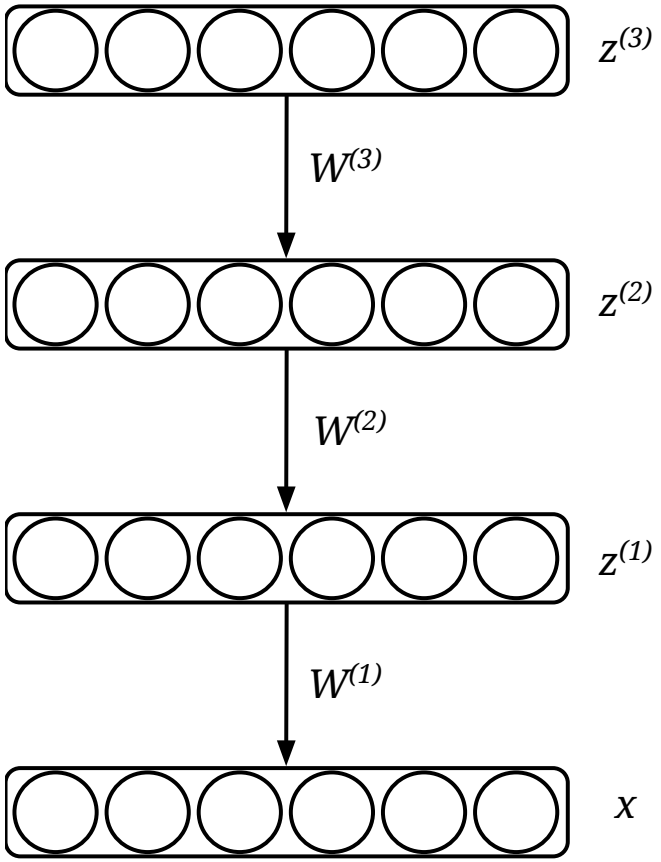
$$z \sim p(z) \qquad x \sim p(x|z)$$

- z is latent/hidden x is observed (image)
- Use θ to denote parameters of the generative model

Deep directed graphical models

$$P(\mathbf{x}) = \sum_h P(\mathbf{x}|\mathbf{h})P(\mathbf{h})$$

- Intractable
- Can't optimize data likelihood directly



Bounding the marginal likelihood

Recall Jenson's inequality: When f is concave, $f(\mathbb{E}[x]) \geq \mathbb{E}[f(x)]$

$$
\begin{aligned}
\log p(x) &= \log \int_z p(x,z) \\
&= \log \int_z q(z) \frac{p(x,z)}{q(z)} \\
&\geq \int_z q(z) \log \frac{p(x,z)}{q(z)} = L(x;\theta,\phi) \qquad \text{(by Jensons inequality)} \\
&= \int_z q(z) \log p(x,z) - \int_z q(z) \log q(z) \\
&= \underbrace{\mathbb{E}_{q(z)}[\log p(x,z)]}_{\text{Expectation of joint distribution}} + \underbrace{\mathrm{H}(q(z))}_{\text{Entropy}}
\end{aligned}
$$

Evidence Lower BOund (ELBO)

- Bound is tight when variational approximation matches true posterior:

$$\begin{aligned}
\log p(x) - L(x;\theta,\phi) &= \log p(x) - \int_z q(z) \log \frac{p(x,z)}{q(z)} \\
&= \int_z q(z) \log p(x) - \int_z q(z) \log \frac{p(x,z)}{q(z)} \\
&= \int_z q(z) \log \frac{q(z)p(x)}{p(x,z)} \\
&= D_{KL}(q(z;\phi)||p(z|x))
\end{aligned}$$

Summary

- Assume existence of $q(z; \phi)$
- Bound $\log p(x; \theta)$ with $L(x; \theta, \phi)$
- Bound is tight when:

$$D_{KL}(q(z; \phi)||p(z|x)) = 0 \iff q(z; \phi) = p(z|x)$$

Learning directed graphical models

- Maximize bound on likelihood of data:

$$\max_{\theta} \sum_{i=1}^{N} \log p(x_i; \theta) \geq \max_{\theta, \phi_1, \ldots, \phi_N} \sum_{i=1}^{N} L(x_i; \theta, \phi_i)$$

- Historically, used different ϕ_i for every data point
  - But we'll move away from this soon..
- $q(z; \phi)$ typically factorized distribution
- For more info see Blei *et al.* (2003)

New method of learning: approximate inference model

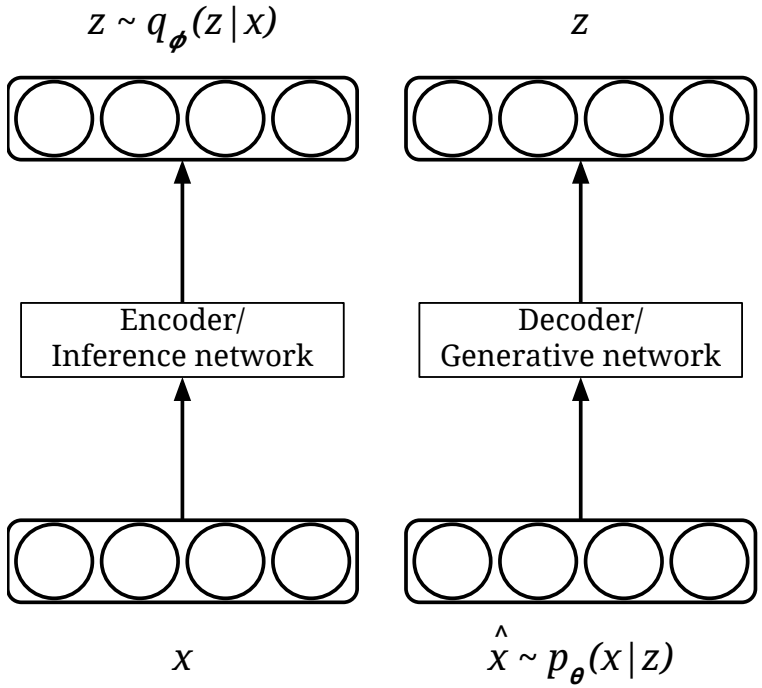
- Instead of having different variational parameters for each data point, fit a conditional parametric function
- The output of this function will be the parameters of the variational distribution $q(z|x)$
- Instead of q(z) we have $q_\phi(z|x)$
- ELBO becomes:

$$L(x;\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x,z)]}_{\text{Expectation of joint distribution}} + \underbrace{\mathrm{H}(q_\phi(z|x))}_{\text{Entropy}}$$

Outline

Variational autoencoder

- *Encoder* network maps from image space to latent space
  - Outputs parameters of $q_\phi(z|x)$
- *Decoder* maps from latent space back into image space
  - Outputs parameters of $p_\theta(x|z)$



[Kingma & Welling (2013)]

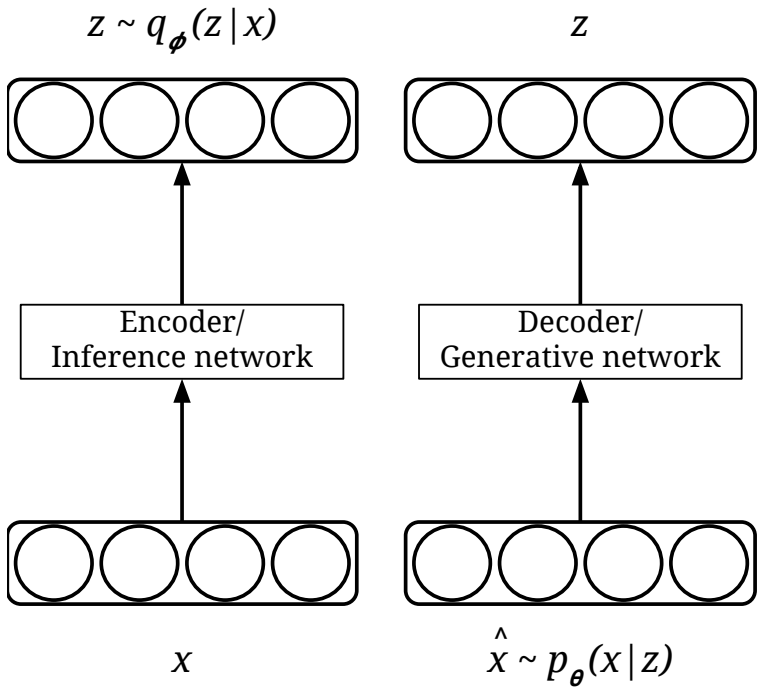
Variational autoencoder

- Rearranging the ELBO:

$$
\begin{aligned}
L(x;\theta,\phi) &= \int_z q_\phi(z|x) \log \int_z \frac{p(x,z)}{q_\phi(z|x)} \\
&= \int_z q_\phi(z|x) \log \int_z \frac{p(x|z)p(z)}{q_\phi(z|x)} \\
&= \int_z q_\phi(z|x) \log p(x|z) + \int_z q_\phi(z|x) \log \frac{p(z)}{q_\phi(z|x)} \\
&= \mathbb{E}_{q(z|x)} \log p(x|z) - \mathbb{E}_{q(z|x)} \log \frac{q(z|x)}{p(z)} \\
&= \underbrace{\mathbb{E}_{q(z|x)} \log p(x|z)}_{\text{Reconstruction term}} - \underbrace{D_{KL}(q(z|x)||p(z))}_{\text{Prior term}}
\end{aligned}
$$

Variational autoencoder

- Inference network outputs parameters of $q_\phi(z|x)$
- Generative network outputs parameters of $p_\theta(x|z)$
- Optimize θ and ϕ jointly by maximizing ELBO:



$$L(x;\theta,\phi) = \underbrace{\mathbb{E}_{q(z|x)} \log p(x|z)}_{\text{Reconstruction term}} - \underbrace{D_{KL}(q(z|x)||p(z))}_{\text{Prior term}}$$

Stochastic gradient variation bayes (SGVB) estimator

- Reparameterization trick : re-parameterize $z \sim q_\phi(h|z)$ as

$$z = g_\phi(x, \epsilon) \text{ with } \epsilon \sim p(\epsilon)$$

- For example, with a Gaussian can write $z \sim \mathcal{N}(\mu, \sigma^2)$ as

$$z = \mu + \epsilon\sigma^2 \text{ with } \epsilon \sim \mathcal{N}(0, 1)$$

[Kingma & Welling (2013); Rezende *et al.* (2014)]

Stochastic gradient variation bayes (SGVB) estimator

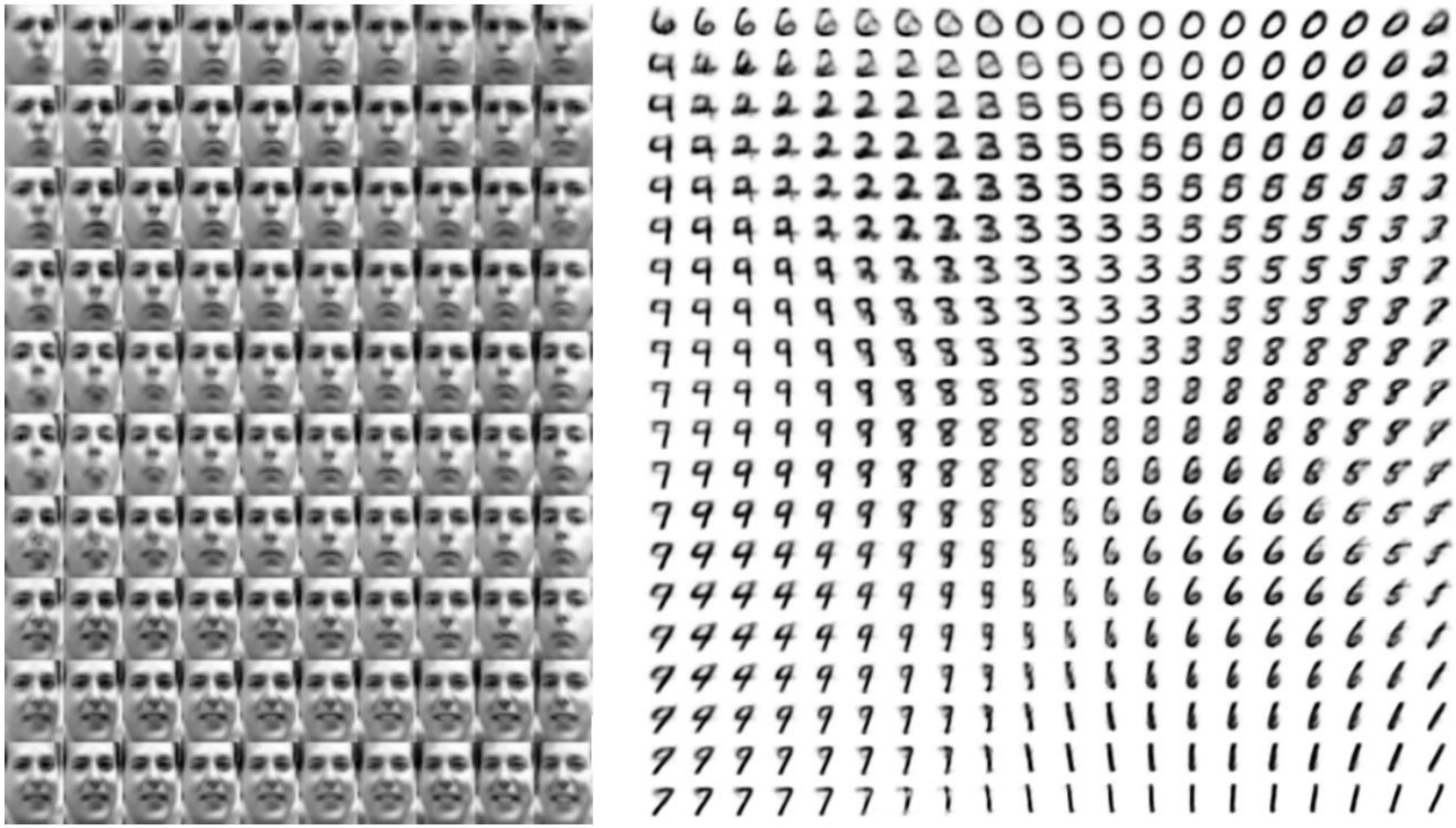
$$L(x;\theta,\phi) = \underbrace{\mathbb{E}_{q(z|x)} \log p(x|z)}_{\text{Reconstruction term}} - \underbrace{D_{KL}(q(z|x)||p(z))}_{\text{Prior term}}$$

- Using reparameterization trick we form Monte Carlo estimate of reconstruction term:

$$\mathbb{E}_{q_\phi(z|x)} \log p_\theta(x|z) = \mathbb{E}_{p(\epsilon)} \log p_\theta(x|g_\phi(x,\epsilon))$$
$$\simeq \frac{1}{L} \sum_{i=1}^{L} \log p_\theta(x|g_\phi(x,\epsilon)) \quad \text{where } \epsilon \sim p(\epsilon)$$

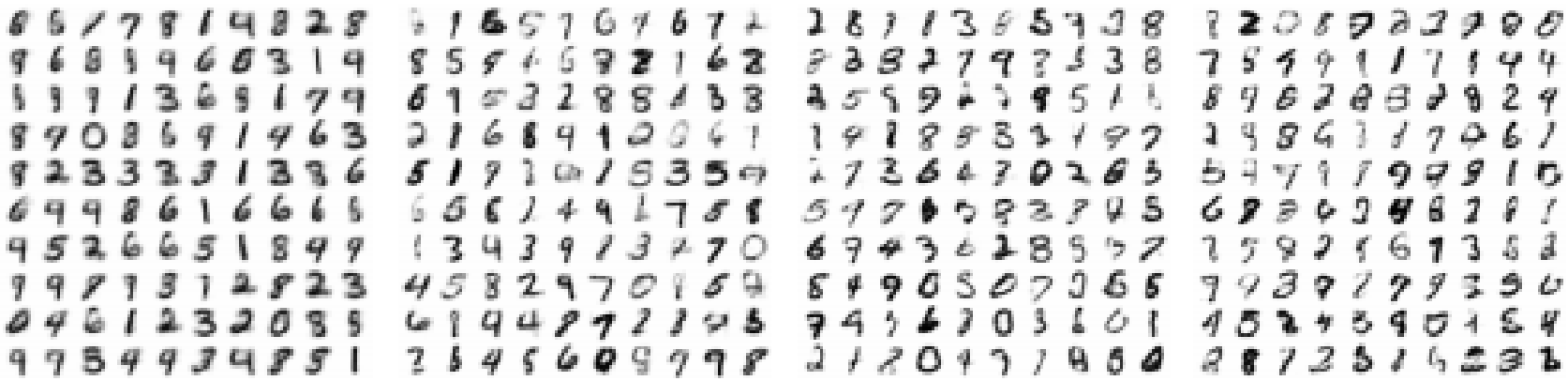
- KL divergence term can often be computed analytically (eg. Gaussian)

VAE learned manifold



[Kingma & Welling (2013)]

VAE samples

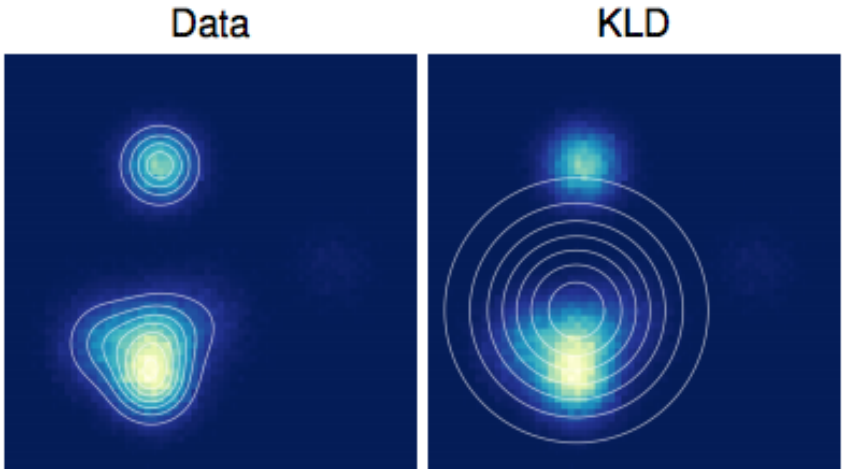


(a) 2-D latent space (b) 5-D latent space (c) 10-D latent space (d) 20-D latent space

[Kingma & Welling (2013)]

VAE tradeoffs

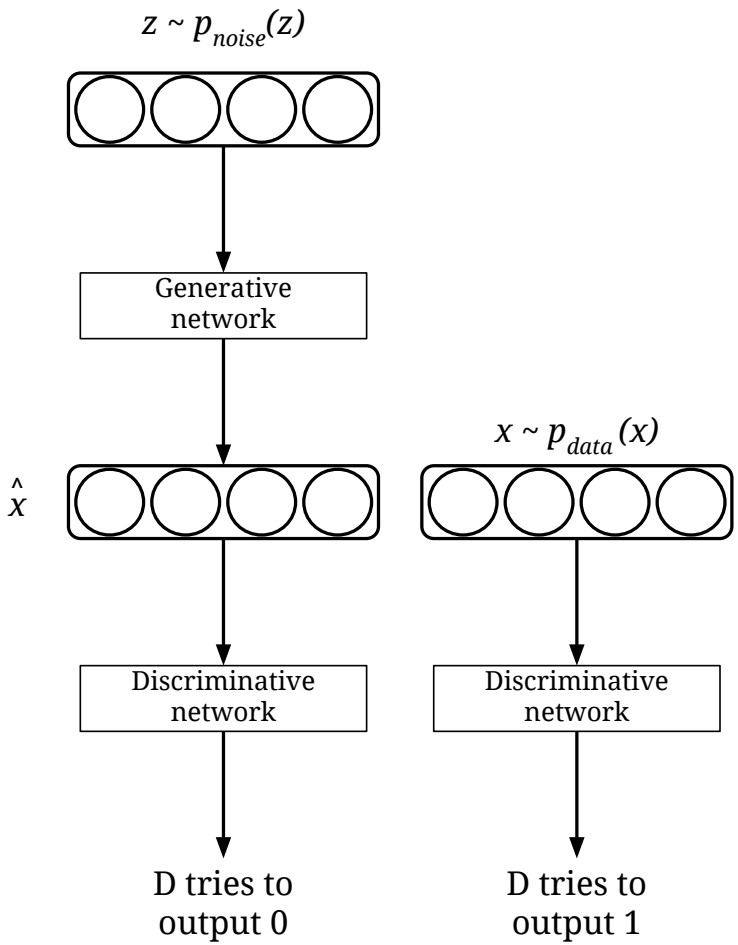
- Pros:
  - Theoretically pleasing
  - Optimizes bound on likelihood
  - Easy to implement
- Cons:
  - Samples tend to be blurry
    - Maximum likelihood minimizes $D_{KL}(p_{data}||p_{model})$



[Theis *et al.* (2016)]

Generative adversarial networks

- Don't focus on optimizing $p(x)$, just learn to sample
- Two networks pitted against one another:
  - Generative model G captures data distribution
  - Discriminative model D distinguishes between real and fake samples



[Goodfellow *et al.* (2014)]

Generative adversarial networks

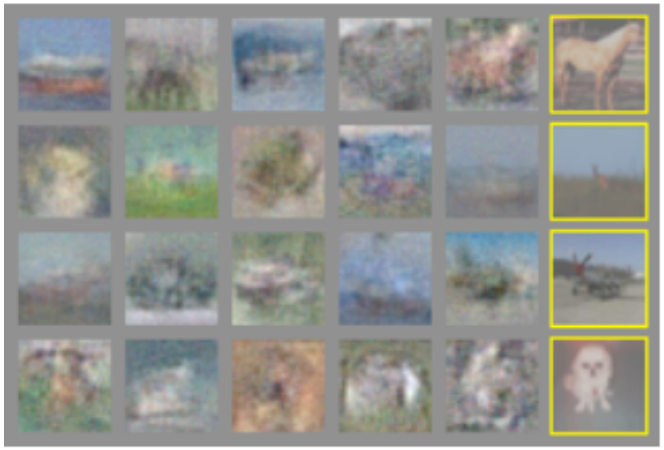
- D is trained to estimate the probability that a sample came from data distribution rather than G
- G is trained to maximize the probability of D making a mistake

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(x)} \log D(x) + \mathbb{E}_{z \sim p_{noise}(z)} \log(1 - D(G(z)))$$

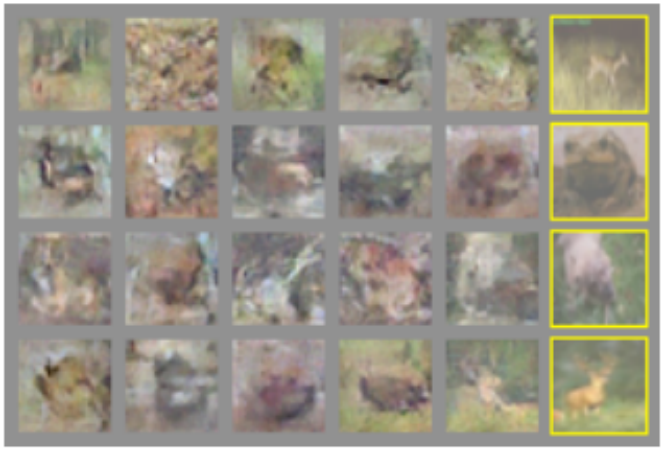
GAN samples

MNIST

TFD



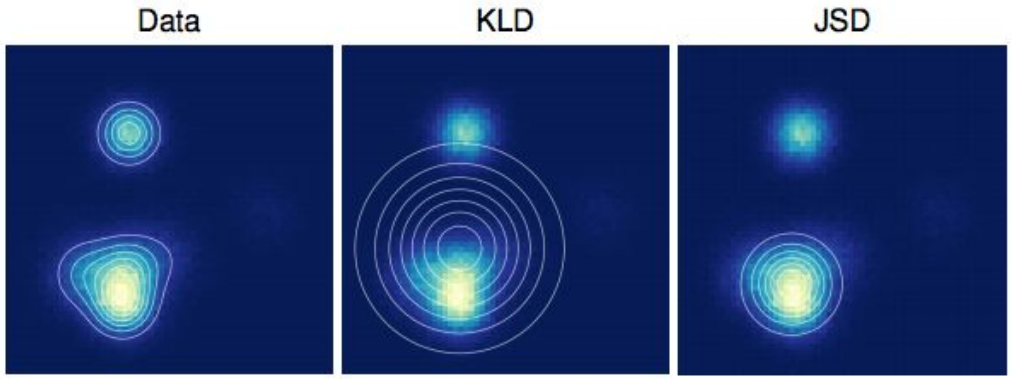
CIFAR-10 (fully connected)



CIFAR-10 (convolutional)

GAN tradeoffs

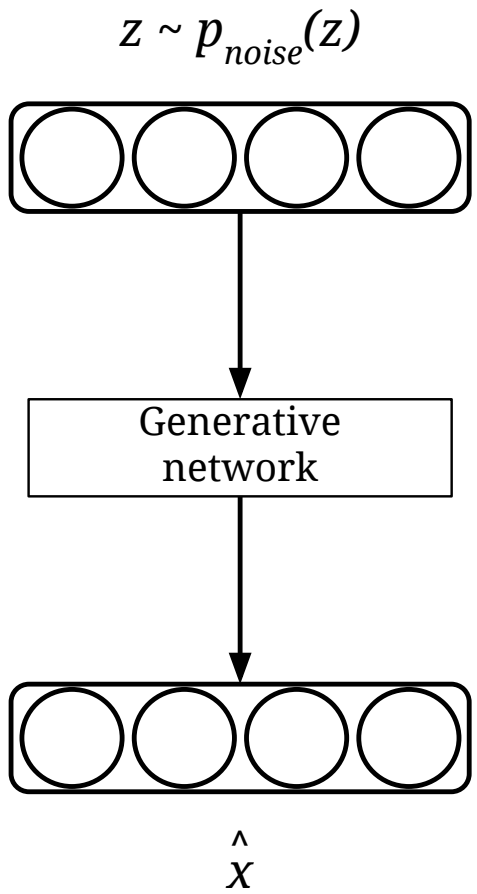
- Pros:
  - Very powerful model
  - High quality samples
- Cons:
  - Tricky to train (no clear objective to track, instability)
  - Can ignore large parts of image space
    - Because approximately minimizing Jenson-Shannon divergence [Goodfellow *et al.* (2014); Theis *et al.* (2016); Huszar (2015)]



[Theis *et al.* (2016)]

Generative moment matching networks

- Same idea as GANs, but different optimization method
- Match moments of data and generative distributions
- Maximum mean discrepancy
  - Estimator for answering whether two samples come from same distribution
- Evaluate MMD on generated samples



[Li *et al.* (2015); Dziugaite *et al.* (2015)]

Generative moment matching networks

$$\mathcal{L}_{MMD^2} = \| \frac{1}{N} \sum_{i=1}^{N} \phi(x_i) - \frac{1}{M} \sum_{j=1}^{M} \phi(x_j) \|^2$$

$$= \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \phi(x_i)^\top \phi(x_{i'}) - \frac{1}{M^2} \sum_{j=1}^{M} \sum_{j'=1}^{M} \phi(x_j)^\top \phi(x_{j'})$$

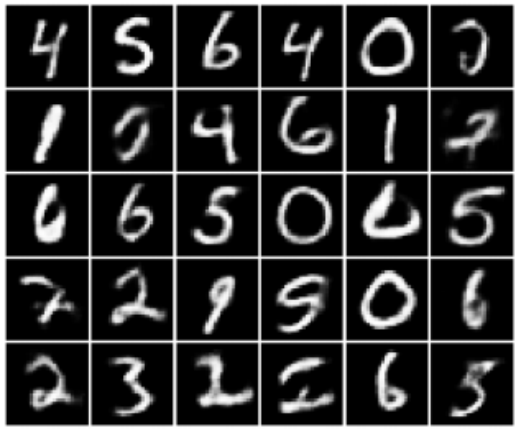
$$- \frac{2}{NM} \sum_{i=1}^{N} \sum j = 1^M \phi(x_i)^\top \phi(x_j)$$

- Can make use of kernel trick
- If ϕ is identity, then matching means
- Complex ϕ can match higher order moments

GMMN samples

(a) GMMN MNIST samples

(b) GMMN TFD samples

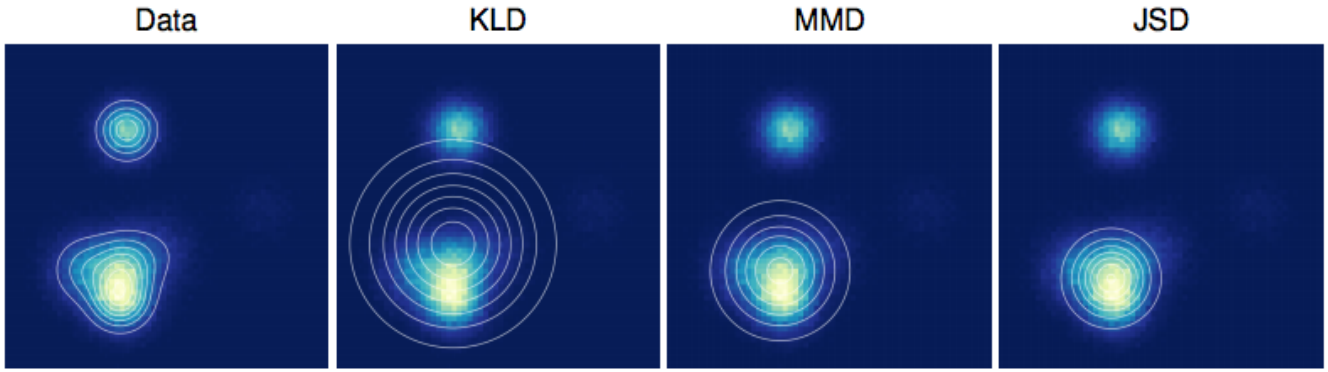


(c) GMMN+AE MNIST samples

(d) GMMN+AE TFD samples

GMMN tradeoffs

- Pros:
  - Theoretically pleasing
- Cons:
  - Batch size very important
  - Samples aren't great (get better when combined with autoencoder)



[Theis *et al.* (2016)]

How to evaluate a generative model?

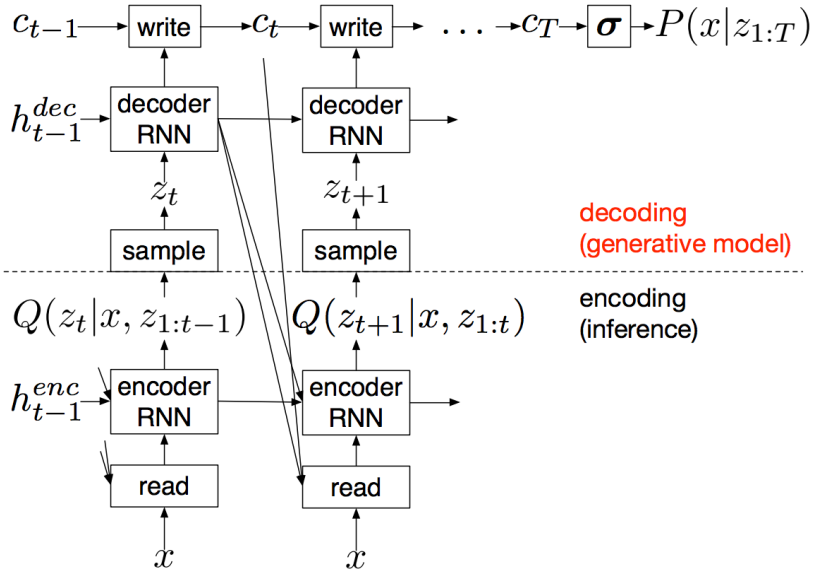
- Log likelihood on held out data
  - Makes sense when goal is density estimation
  - Many approaches don't have tractable likelihood or it isn't explicitly represented
  - Have to resort to Parzen window estimates [Breuleux *et al.* (2009)]
- Quality of samples
  - But a lookup table of training images will succeed here...
- Best: evaluate in context of particular application
- See Theis *et al.* (2016) for more details

Outline

DRAW: Deep Recurrent Attentive Writer

Basic idea:

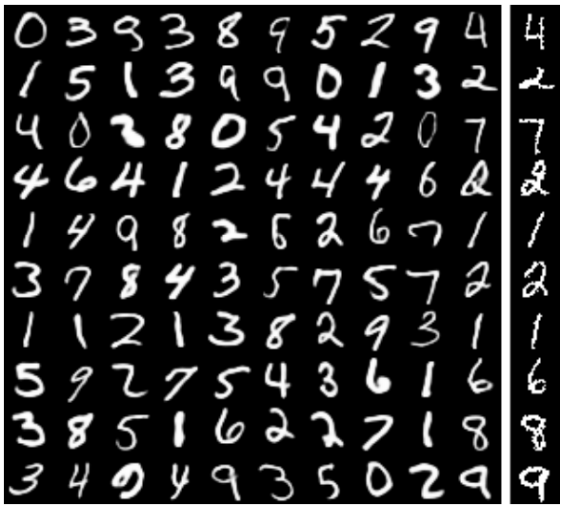
- Iteratively construct image
- Observe image through sequence of glimpses

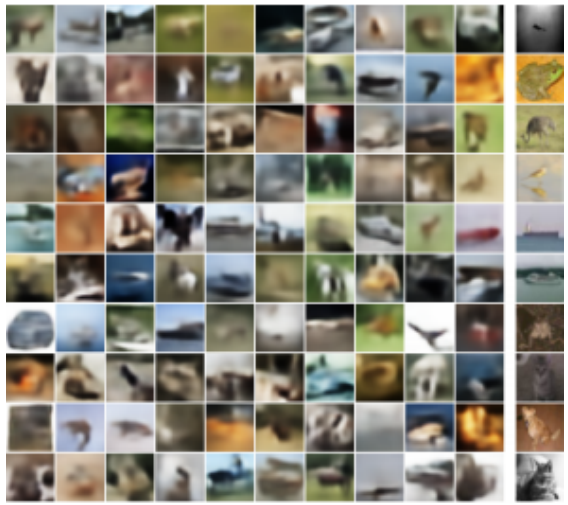


[Gregor *et al.* (2015)]

- Recurrent encoder and decoder
- Optimizes variational bound
- Attention mechanism determines:
  - Input region observed by encoder
  - Output region modified by decoder

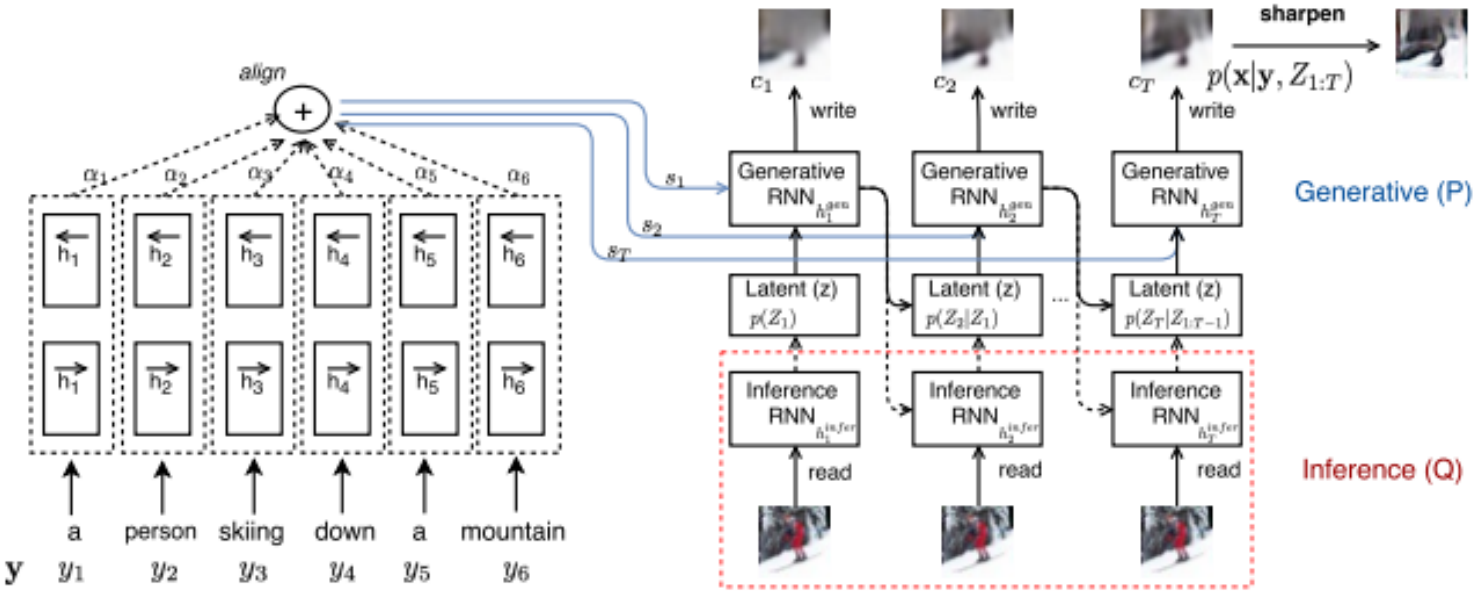
DRAW samples





Generating images from captions

- Language model: bidirectional RNN
- Image model: conditional DRAW network
- Image sharpening with Laplacian pyramid adversarial network



[Mansimov *et al.* (2016)]

Generating images from captions

A yellow school bus parked in a parking lot.

A red school bus parked in a parking lot.

A green school bus parked in a parking lot.

A blue school bus parked in a parking lot.

The decadent chocolate desert is on the table.

A bowl of bananas is on the table.

A vintage photo of a cat.

A vintage photo of a dog.

[Mansimov *et al.* (2016)]

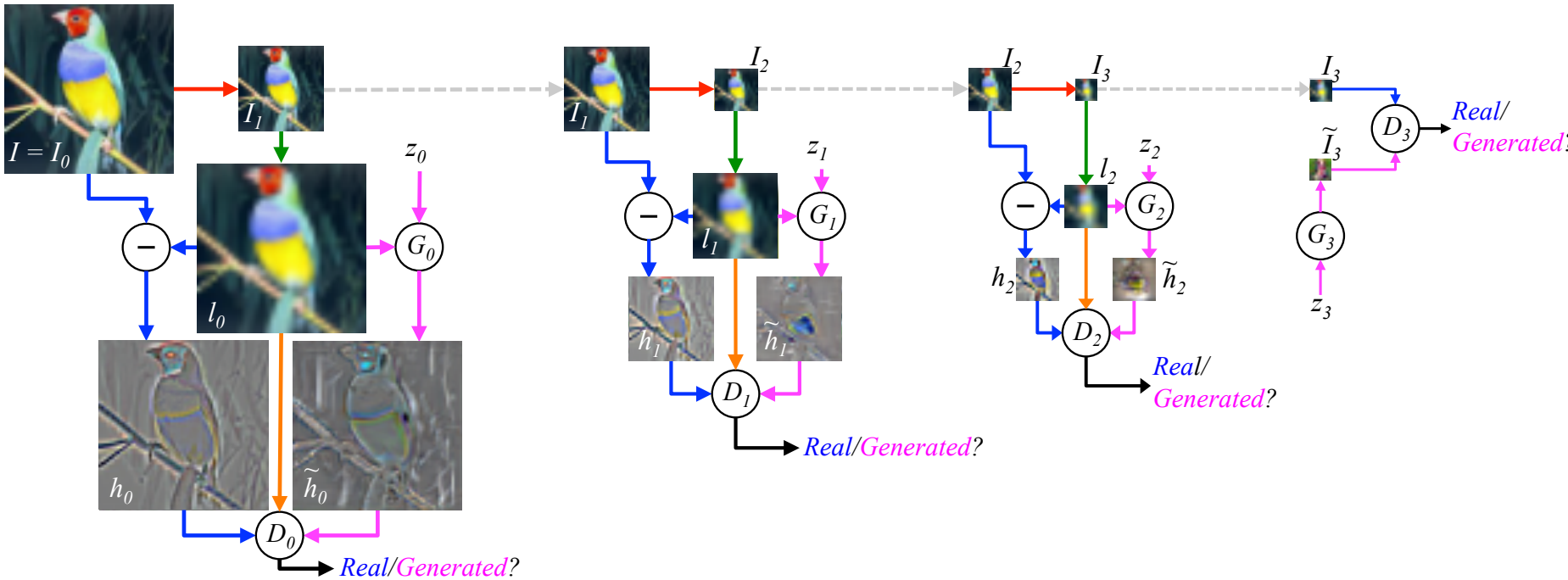
Laplacian pyramid of adversarial networks

- Difficult to generate large images in one shot
- Break problem up into sequence of manageable steps
- Samples drawn in coarse-to-fine fashion

- Each scale is a convnet trained using GAN framework

[Denton *et al.* (2015)]

LAPGAN training procedure

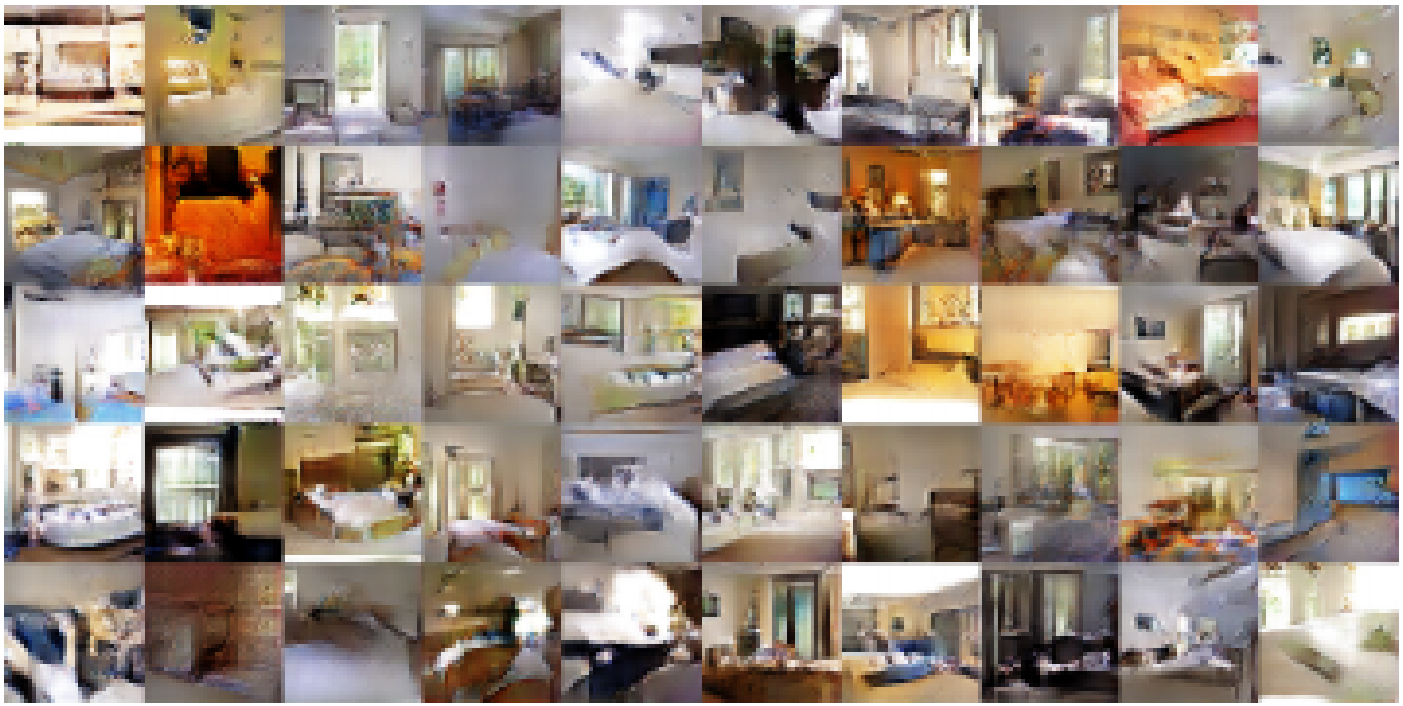


LAPGAN samples

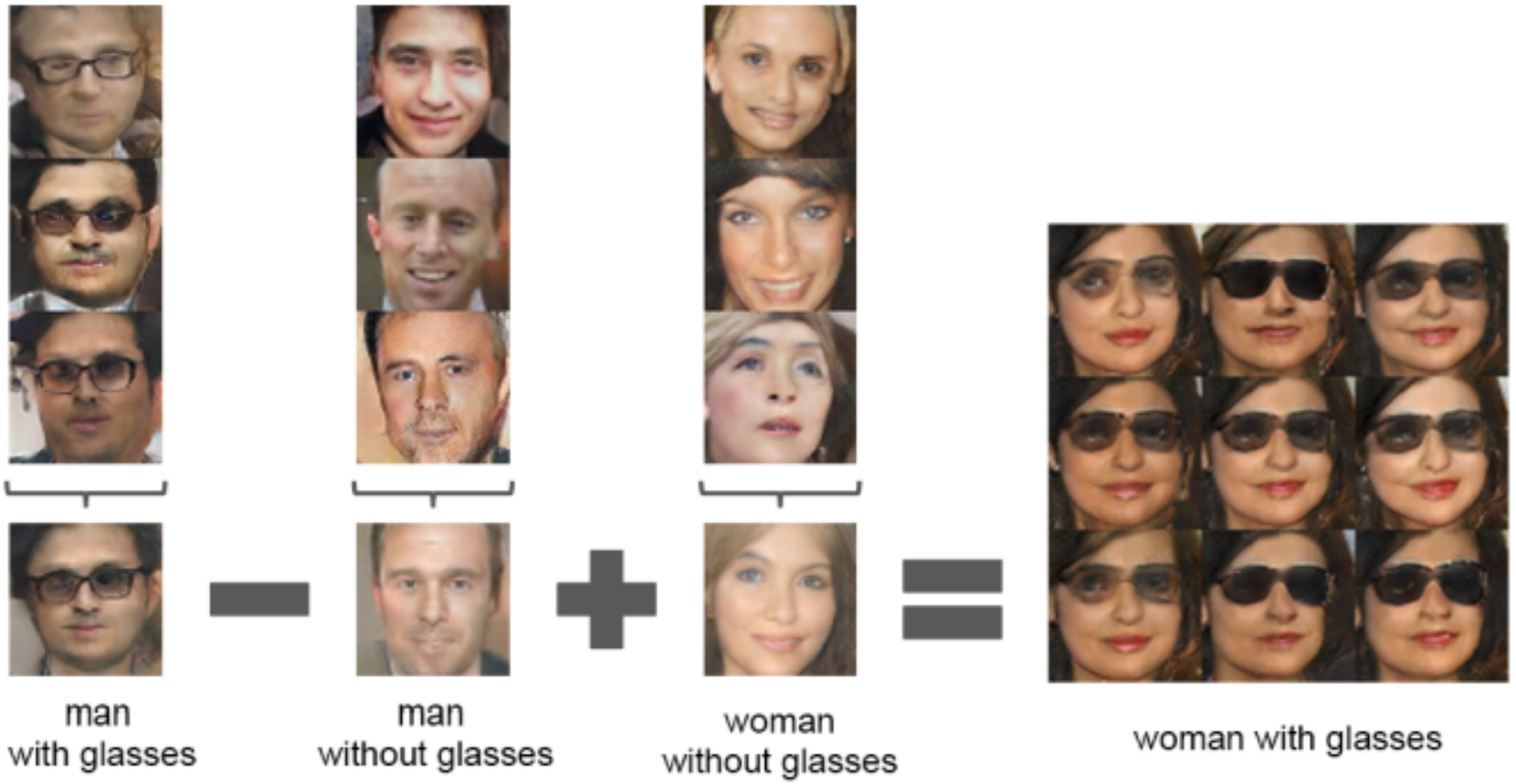
LAPGAN samples

Deep convolutional generative adversarial networks (DCGAN)

- Radford *et al.* (2016) propose several tricks to make GAN training more stable

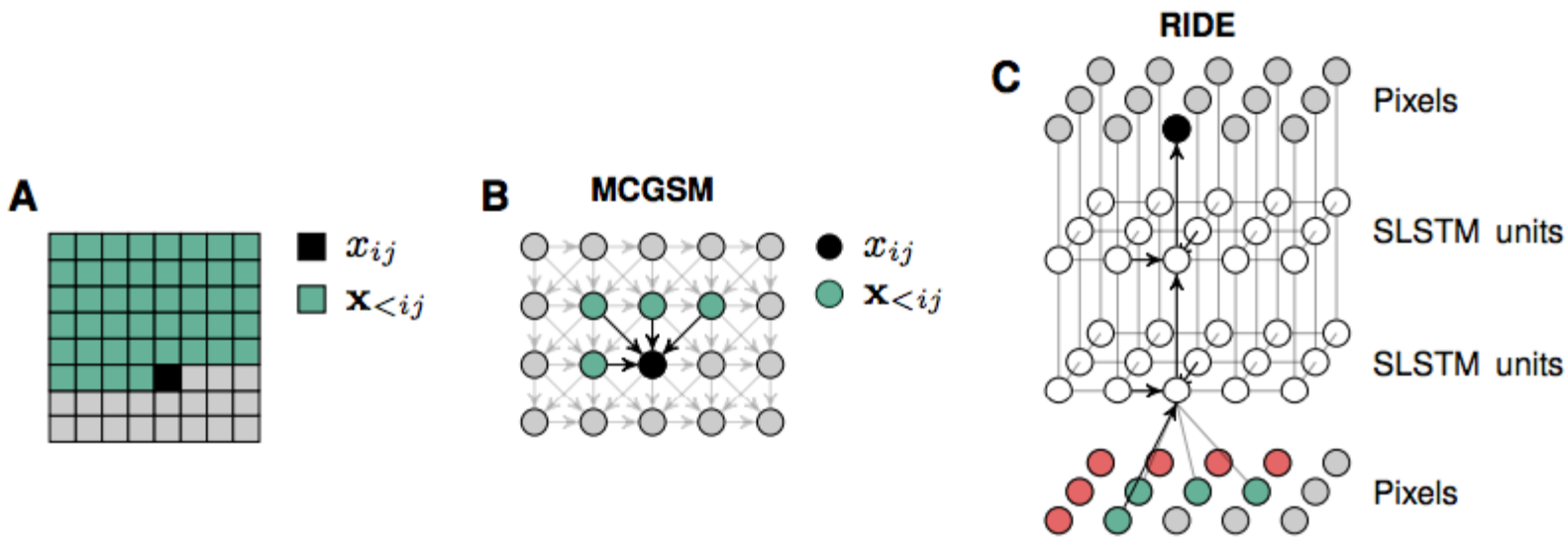


DCGAN vector arithmetic



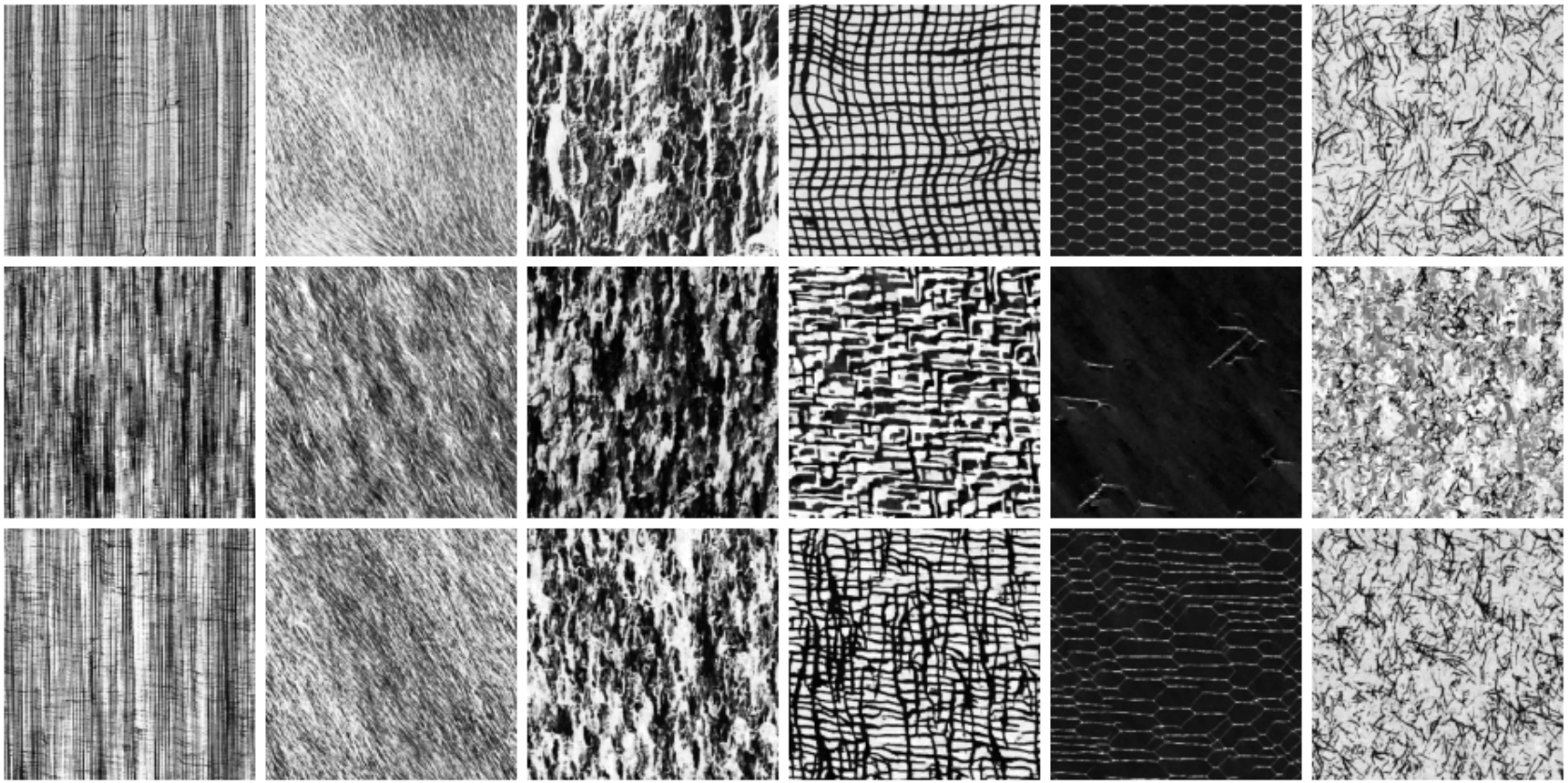
Texture synthesis with spatial LSTMs

- Two dimensional LSTM
- Sequentially predict pixels in an image conditioned on previous pixels



[Theis & Bethge (2015)]

Texture synthesis with spatial LSTMs



[Theis & Bethge (2015)]

Pixel recurrent neural networks

- Sequentially predict pixels in an image conditioned on previous pixels
- Uses spatial LSTM

[van den Oord *et al.* (2016)]

References I

Blei, David M., Ng, Andrew Y., & Jordan, Michael I. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research.*

Breuleux, O., Bengio, Y., & Vincent, P. 2009. Unlearning for better mixing. *Technical report, Universite de Montreal.*

Denton, Emily, Chintala, Soumith, Szlam, Arthur, & Fergus, Rob. 2015. Deep generative image models using a laplacian pyramid of adversarial networks. *In: NIPS.*

Dziugaite, Gintare Karolina, Roy, Daniel M., & Ghahramani, Zoubin. 2015. Training generative neural networks via Maximum Mean Discrepancy optimization. *In: UAI.*

References II

Goodfellow, Ian J., Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron C., & Bengio, Yoshua. 2014. Generative adversarial networks. *In: NIPS*.

Gregor, Karol, Danihelka, Ivo, Graves, Alex, & Wierstra, Daan. 2015. DRAW: A Recurrent Neural Network For Image Generation. *CoRR*, **abs/1502.04623**.

Huszar, F. 2015. How (not) to train your generative model: schedule sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*.

Kingma, Diederik P, & Welling, Max. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

References III

Li, Yujia, Swersky, Kevin, & Zemel, Richard. 2015. Generative Moment Matching Networks. *In: ICML*.

Mansimov, Elman, Parisotto, Emilio, Ba, Jimmy, & Salakhutdinov, Ruslan. 2016. Generating Images from Captions with Attention. *In: ICLR*.

Radford, Alec, Metz, Luke, & Chintala, Soumith. 2016. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *In: ICLR*.

Rezende, Danilo Jimenez, Mohamed, Shakir, & Wierstra, Daan. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Theis, Lucas, & Bethge, Matthias. 2015. Generative Image Modeling Using Spatial LSTMs. *In: NIPS*.

References IV

Theis, Lucas, van den Oord, Aaron, & Bethge, Matthias. 2016. A note on the evaluation of generative models. *In: ICLR*.

van den Oord, Aaron, Kalchbrenner, Nal, & Kavukcuoglu, Koray. 2016. Pixel Recurrent Neural Networks. *In: ICML*.